Attorney Docket No. 1793.1189

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Hai JIN et al.

Application No.: 10/763,422 Group Art Unit: Unassigned

Filed: January 26, 2004 Examiner: Unassigned

For: VIDEO SPLITTING AND DISTRIBUTED PLACEMENT SCHEME FOR CLUSTERED VIDEO SERVERS

SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN APPLICATION IN ACCORDANCE WITH THE REQUIREMENTS OF 37 C.F.R. § 1.55

Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Sir:

In accordance with the provisions of 37 C.F.R. § 1.55, the applicant(s) submit(s) herewith a certified copy of the following foreign application:

Chinese Patent Application No(s). 03118543.6

Filed: January 25, 2003

It is respectfully requested that the applicant(s) be given the benefit of the foreign filing date(s) as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. § 119.

Respectfully submitted,

STAAS & HALSEY LLP

Date: 4/20/04 By: _____

Michael D. Stein
Registration No. 37,240

1201 New York Ave, N.W., Suite 700
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501

证　　明

本证明之附件是向本局提交的下列专利申请副本

申　请　日：　2003.01.25

申　请　号：　03118543.6

申请类别：　发明

发明创造名称：　基于集群视频服务器的节目源分片分布式存储方法

申　请　人：　华中科技大学

发明人或设计人：　金海、廖小飞、胡琼华、庞丽萍

中华人民共和国
国家知识产权局局长　王景川

2004 年 3 月 22 日

权 利 要 求 书

1．一种基于集群视频服务器的节目源分片分布式存储方法，顺序包括如下步骤：

(1) 执行流媒体网络报文定义步骤，给出网络报文的结构。同时执行流媒体分布式控制文件定义步骤和分片文件定义步骤，定义媒体分布式控制文件的结构和分片文件的总体结构；

(2) 执行流媒体源文件信息获取步骤，获取流媒体源文件的基本信息，同时执行用户需求信息与获取步骤，接受用户的基本分片要求；

(3) 执行分片文件放置策略定义步骤，得到用户所定义的分片文件放置要求；

(4) 执行流媒体文件分析与分片任务列表生成步骤，分析用户所定义的分片文件放置要求和媒体源文件，得到分片任务列表以及相应的控制信息文件；

(5) 执行分片任务执行步骤，根据源文件分片数目建立多个处理线程，每个线程执行一个分片任务；

(6) 分片传递存储步骤，依据分片文件放置策略定义步骤所定义的放置要求，把相应的分片文件传递到相应的的存储节点上。

2．如权利要求 1 所述的基于集群视频服务器的节目源分片分布式存储方法，其特征如下：

(1) 所述流媒体网络报文定义步骤定义遵从国际实时传输协议的流媒体数据报文主体部分的格式，包括媒体类型头、序列号、时间戳、同步源和主体数据；

(2) 所述流媒体源文件分布式控制文件定义步骤包括 Index 文件和 SDP 文件，Index 文件包括播放任务列表、电影资源文件名、电影资源空间大小、电影资源时间长度、电影资源分片数目、电影资源热点度组成，SDP 文件包括媒体类型编号、电影资源包含的媒体流的数目、电影资源的时间长度、客户会话的唯一表示码；

(3) 所述分片文件定义步骤定义分片文件结构，包括分片文件头部、媒体流信息头、媒体流数据包；

(4) 所述流媒体源文件信息获取步骤依据媒体源文件的逻辑时间特点，读取媒体源文件的逻辑时间单元，对于每一个逻辑时间单元获取其头部的时间信息、媒体流的个数，如此循环执行，直至完成所有逻辑时间单元，得到总的播放的时间长度、空间大小，并依据该媒体格式的规定，获取媒体格式类型 ID；

(5) 所述用户需求信息与获取步骤包括分片时间要求与获取步骤和分片放置策略要求与获取步骤；

(6) 所述分片文件放置定义步骤，包含数据放置策略选择项、电影资源热点级别选择项以及可供调用的数据放置实现算法；

(7) 所述流媒体文件分析与分片任务列表生成步骤，通过获得分片文件放置策略信息，得到用户定义的分片文件放置要求，同时分析媒体源文件以寻找每一个分片文件在源文件中的空间时间偏移量以及网络报文的序列号范围，在次基础上，产生分片任务列表；

(8) 所述分片任务执行步骤需先读取 Index 文件，取得电影资源分片数目，以此数目建立多个处理线程，其次继续读取 Index 文件，获得播放任务列表，把该列表每个表项的内容交给相应的处理线程，从而建立分片任务，然后对于任意一个分片任务，在电影资源文件中定位所规定的空间位置，寻找媒体数据的解码单元 Pack 标志，当遇到此标志时，就读出这个数据单元，并按照流媒体网络报文定义步骤的规定，分割成若干个流媒体网络数据包。并写入相应的分片文件；循环操作，直至所有相关的数据单元都完全读完。

说　　明　　书

一种基于集群视频服务器的节目源分片分布式存储方法

技术领域

本发明属于信息处理与存储方法，其具体涉及并行分布式处理技术与视频处理技术相结合的交叉领域，适用于集群的视频服务器节目源分片分布存储方法。

背景技术

随着宽带网络、声音图像数字编码技术的发展，流媒体、富媒体的应用越来越普及。在这些应用中重要的基础设施是视频服务器。为满足应用的需要，视频服务器对计算机性能有着特殊的要求。媒体服务有数据量大、实时要求高等特点，一般的单机服务器由于CPU、内存、网络和硬盘的瓶颈，通常只能服务几十个用户，不能满足媒体服务应用；而国外高性能的服务器则价格非常昂贵。

集群技术因其高度可扩展性和廉价的成本为视频服务器的实现提供了一个良好的技术基础。集群系统的显著技术特点是存储单元的分散性、单个节点的自治性以及控制的可集中性。基于这样的特点，集群架构的视频服务器越来越成为人们关注研究的一个领域。

基于集群的视频服务器，关键技术之一是实现电影资源（节目源）的分布式存储。每一部电影都需要按照一定的方法进行切割分片，从而能够分散地存储在集群的每一个存储节点上。因此寻找高效率、高可用、高通用性的电影资源的分片分布式存储的方法成为亟待解决的一个重要的问题。

电影资源的存储有两种典型的解决方法。一种是从电影资源本身的播放时间入手，按照等长的电影播放时间把一个电影文件切割成多个分片（P. Shenoy, P. Goyal, and H. M. Vin, “Issues in Multimedia Server Design”, ACM Computing Surveys, Vol. 27, No. 4, pp. 636-639, December 1995）；另一种方法则是从电影资源的空间大小着手，按照大小相等的电影存储空间将电影文件切割成为多个

小片（P. Shenoy and H. M. Vin. “Efficient Striping Techniques for Multimedia File Servers”, Proceedings of the 7th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'97), pp. 25-36, May 1997）。前者实质是按照相等时间长度分片，后者则是按照相等空间大小分片。从特点上来讲，这两种方都存在着一些不能忽视的问题。相等时间分片的策略实现起来比较复杂。因为一方面，每一种媒体格式从媒体数据的压缩比率到存储格式都是完全不同的；另一方面，即使是同一种媒体格式的文件中，不同阶段的场景变换完全不可预测，导致画面的丰富程度不同，因此相等时间长度的媒体数据的存储空间大小不同。这两个原因导致该方法实现的复杂度。其次，该方法不具备通用性，对每一种媒体格式的处理因其格式标准的不同将是完全不同的。对于相等空间大小的策略用于媒体访问的一些随机命令将不能够方便的获得处理。因为，既然是按照空间大小分片，就可能存在一个媒体电影中的不同媒体流在某一个分片中不能同步，最终导致电影播放的质量问题。其次，这种方法导致了需要在跨越两台节点的相继分片媒体文件上的同步操作。这种操作一方面降低了系统处理多个同时在线流的性能，另一方面也使视频服务器媒体数据接口模块的实现难度加剧，控制节点与数据节点的内部通信量加大。

以上两种方法（按时间分片的方法与按空间大小分片的方法）实现的模块结构都包含如下四个部分：用户需求信息与获取模块、分片文件定义模块、流媒体文件信息获取模块、分片执行模块。这些模块在这两种方法中的意义是有区别的。如用户需求信息与获取模块，其主要功能是获取用户对于分片的要求，按时间分片的方法中，是获取每一个分片的时间长度；但是在按空间大小分片的方法中，是获取每一个分片的空间大小。分片文件定义模块在这两种方法之间的差别也是如此，在按照时间分片的方法中，流媒体文件信息获取模块主要获取每一个时间点所对应的空间位置；而在按照空间大小分片的方法中，工作简单一些，主要是获取对应大小的空间位置。这两种方法在分片执行模块是一样的。

发明内容

本发明针对现有的集群视频服务器的电影资源分片分布式存储系统的不足，提出了一种基于集群视频服务器的节目源分片分布式存储方法。

本发明采用的是一种按照时间长度和按照空间长度相结合的新的分片方法；并采用分片分布式存储的方法，以更充分发挥集群系统的并行处理的特点。

为实现上述发明目的，本发明的基于集群视频服务器的分片分布式存储方法顺序包括如下步骤：

(1) 执行流媒体网络报文定义步骤，给出网络报文的结构。同时执行流媒体分布式控制文件定义步骤和分片文件定义步骤，定义媒体分布式控制文件的结构和分片文件的总体结构；

(2) 执行流媒体源文件信息获取步骤，获取流媒体源文件的基本信息，同时执行用户需求信息与获取步骤，接受用户的基本分片要求；

(3) 执行分片文件放置策略定义步骤，得到用户所定义的分片文件放置要求；

(4) 执行流媒体文件分析与分片任务列表生成步骤，分析用户所定义的分片文件放置要求和媒体源文件，得到分片任务列表以及相应的控制信息文件；

(5) 执行分片任务执行步骤，根据源文件分片数目建立多个处理线程，每个线程执行一个分片任务；

(6) 分片传递存储步骤，依据分片文件放置策略定义步骤所定义的放置要求，把相应的分片文件传递到相应的的存储节点上。

所述的基于集群视频服务器的节目源分片分布式存储方法，其进一步的特征如下：

(1) 所述流媒体网络报文定义步骤定义遵从国际实时传输协议的流媒体数据报文主体部分的格式，包括媒体类型头、序列号、时间戳、同步源和主体数据；

(2) 所述流媒体源文件分布控制文件定义步骤包括 Index 文件和 SDP 文件，Index 文件包括播放任务列表、电影资源文件名、电影资源空间大小、电影资源时间长度、电影资源分片数目、电影资源热

点度组成，SDP 文件包括媒体类型编号、电影资源包含的媒体流的数目、电影资源的时间长度、客户会话的唯一表示码；

(3) 所述分片文件定义步骤定义分片文件结构，包括分片文件头部、媒体流信息头、媒体流数据包；

(4) 所述流媒体源文件信息获取步骤依据媒体源文件的逻辑时间特点，读取媒体源文件的逻辑时间单元，对于每一个逻辑时间单元获取其头部的时间信息、媒体流的个数，如此循环执行，直至完成所有逻辑时间单元，得到总的播放的时间长度、空间大小，并依据该媒体格式的规定，获取媒体格式类型 ID；

(5) 所述用户需求信息与获取步骤包括分片时间要求与获取步骤和分片放置策略要求与获取步骤；

(6) 所述分片文件放置定义步骤，包含数据放置策略选择项、电影资源热点级别选择项以及可供调用的数据放置实现算法；

(7) 所述流媒体文件分析与分片任务列表生成步骤，通过获得分片文件放置策略信息，得到用户定义的分片文件放置要求，同时分析媒体源文件以寻找每一个分片文件在源文件中的空间时间偏移量以及网络报文的序列号范围，在次基础上，产生分片任务列表；

(8) 所述分片任务执行步骤需先读取 Index 文件，取得电影资源分片数目，以此数目建立多个处理线程，其次继续读取 Index 文件，获得播放任务列表，把该列表每个表项的内容交给相应的处理线程，从而建立分片任务，然后对于任意一个分片任务，在电影资源文件中定位所规定的空间位置，寻找媒体数据的解码单元 Pack 标志，当遇到此标志时，就读出这个数据单元，并按照流媒体网络报文定义步骤的规定，分割成若干个流媒体网络数据包。并写入相应的分片文件；循环操作，直至所有相关的数据单元都完全读完。

本发明综合运用了媒体格式标准综合分析技术、流媒体数据切割技术、流媒体实时传输协议实现技术，结合了传统的按照时间长度和按照空间长度进行分片的两种方法的优点，并充分挖掘了集群系统的特点。具体而言，本发明具有以下特点。

(1) 高通用性

本系统理论上完全支持所有现存的媒体格式，并不依赖于某种特

定的媒体格式。尽管不同的媒体格式存在不同的编码速率以及存储格式，但是它们无一例外的都有一个共同的特点，即都是以时间为线索来组织媒体数据，并存储在介质上的。因此，本系统设计与媒体格式无关的上层包装，用于封装具体的、变化多样的媒体数据。对于视频服务器本身而言，它最终所看到的媒体数据都是格式一致相同的。

(2) 高可用性

由本发明所创建的分片分布式媒体数据的存储系统是稳定可靠的，并以一种单一的格式标准与视频服务器上层的读写接口交互，使得文件读写接口显得简单、稳定、可靠。由于视频服务器端底层的读写接口简单明了，明显地降低了整个服务器的出错概率，增加了整个服务器的可靠性。

(3) 高效率

采用分片分布式存储技术使视频服务器具有高效率。所有影片进入本存储系统之前，都必须接受本发明的文件切割操作，分割后的文件分片分布式存储在多个节点上。文件分片过程具有高效率、高稳定的特点，无需担心它的速度。

(4) 明显提升整个分布式视频服务器的性能

本发明通过文件切割操作；提供对所有存储的媒体文件而言的单一标准结构，使视频服务器底层的文件接口模块读写媒体数据的效率大为提升。媒体数据从文件读入到发送至网络的整个过程中，中转缓存的次数减少了三分之一；从而节省了服务器节点的系统资源。同时，媒体数据包的大小基本相等，也使得网络带宽利用合理，不会出现因占用带宽大小急剧变化，从而造成网络资源的浪费。

(5) 有利于视频服务器分布式录制功能的实现

视频服务器作为交互式服务器，录制功能是不可缺少的。该功能广泛应用于远程教育、现场直播等领域。实现分布式录制功能的一个技术难点是如何在多个节点上分布存储录制的媒体数据；同时支持在线客户的即时点播。本发明设计的媒体文件的单一标准格式有利于数据的分布式存储、分布式的随机访问，能有效地解决分布式录制功能的上述难点。

(6) 有利于流媒体客户端播放器的设计

流媒体客户端播放器设计最大的难点在于如何组装网络流媒体数据包。考虑到媒体数据包含了多个流，因此在客户端接收数据之后，组装报文、同步多个流是技术难点。本发明设计的流媒体网络包的组成恰恰省却了同步工作，能方便地将多个流合成为一个单一流来传送，因此报文的组装也相对简单。

附图说明

图 1　本发明流程框图
图 2　本发明所包括的各个步骤示意图
图 3　流媒体网络数据报文的结构
图 4　Index 文件结构
图 5　SDP 文件结构
图 6　分片文件结构
图 7　分片文件的头部信息数据结构
图 8　媒体流的信息头部的数据结构
图 9　分片文件中媒体流数据包的数据结构
图 10 流媒体文件分析及分片任务列表生成步骤处理流程图
图 11 分片任务列表的数据结构
图 12 分片任务执行步骤处理流程图

具体实施方式

下面结合附图对本发明作进一步说明。

媒体源文件分片处理过程见图 1。本系统首先预定义分片文件结构、流媒体网络报文结构、媒体数据分布式控制文件结构以及分片分布式的放置策略等一系列的必要信息；然后执行流媒体文件信息获取模块，以获得源文件的基本信息，为后面的分片工作做准备。紧接着进行流媒体文件分析，生成分片任务列表并执行任务列表。

下面，根据图 1 的媒体源文件分片工作流程示意图，详细地介绍媒体源文件分片处理过程。

媒体源文件分片处理首先调用流媒体网络报文定义步骤，给出网

络报文的基本定义；基于这个网络报文结构，同时执行流媒体分布式控制文件定义步骤以及分片文件定义步骤，定义媒体分布式控制信息文件的结构和分片文件的总体结构。接着，调用媒体源文件信息获取模块，以取得媒体源文件的基本信息，为后面的工作做准备；同时，接受用户的基本分片要求。用户的分片要求主要依据于一些关键的参数选择；有两种方式：一种是依据分片的数目，即用户可以自定义要求分多少个分片文件；一种是规定分片的播放时间长度 Clip_Time，可以获知媒体文件的总的播放时间，于是可以得到总的分片数目。

紧接着，执行分片文件放置策略定义步骤，得到用户所定义的分片文件放置要求；然后执行流媒体文件分析与分片任务列表生成步骤，得到分片任务列表以及相应的控制信息文件。

得到分片任务列表之后，就执行分片任务执行步骤。即创建多线程来操作，每个线程执行一个分片任务。该任务首先读取这个分片任务数据结构中的信息；然后，打开文件，定位到分片任务结构中所指明的空间偏移位置，开始读取媒体数据的解码单元 Pack，每读一个 Pack 时，依据流媒体网络报文结构定义模块所说明的方式，生成若干个流媒体网络报文，又称为 RTP 报文。循环操作，直至所有相关的解码单元都完全读完，那么本分片文件就形成了。

最后，执行分片传递存储步骤。把得到的每一个分片文件都按照分片文件放置策略定义步骤所定义的要求存储到相应的集群的存储节点上。这样，一个分片任务就最终完成了。在这个分片任务执行中，定时监测它的完成百分比，显示给用户浏览；并记载它的调度时间信息并判断是否成功，以返回给用户决策。

本发明的总体结构如图 2 所示，包括用户需求信息与获取步骤、文件定义步骤、流媒体文件信息获取步骤、信息分析与分片任务列表生成步骤、分片任务执行步骤、分片传递存储步骤。用户需求信息与获取步骤又包括：分片时间要求与获取、分片放置策略要求与获取两个步骤。文件定义步骤包括：流媒体网络报文定义、流媒体分布式控制文件定义、分片文件放置策略定义、分片文件定义四个步骤。

流媒体网络报文定义步骤

本步骤定义流媒体网络报文的组成。在本系统中，流媒体网络报文的基本结构遵从流媒体协议的具体标准。业界已经定义了流媒体的几个国际通行的标准协议，RTSP（实时流控制协议）用于客户端与服务器的命令交互控制；RTP/RTCP（实时传输协议以及实时传输的控制协议）用于规定、控制网络流媒体数据报文；SDP（会话描述协议）用于定义客户端与服务器的连接描述。本系统详细定义了遵从国际 RTP 协议的流媒体数据报文的主体部分的特殊格式。如图 3 所示，Decoding Unit（解码单元）是一个可以被解码器接收、并进行解码的数据单元。这个单元的空间大小基本固定，其大小记为：Decoding_Uint_Size。以 MPEG-1 为例，其大小大约为 2000 多个字节，依据不同的压缩比率稍有不同。从图中可以看到，本系统固定报文主体的大小，记为：Packet_Payload_Size。现在，问题的关键是把每一个解码单元分割成多个报文来存储，显然，大部分报文的大小仍然依赖其规定的报文主体固定大小；另一小部分报文大小将依赖于解码单元分割出来多余的一部分数据的大小。由于解码单元是固定大小的，因此这部分的报文大小也是固定的。

解码单元被分割之后，应保证在客户端能够被无一例外的组装起来。本步骤块提供了必要的信息来保证这一点。RTP 标准协议规定协议报文的头部带有时间戳、序列号、媒体标号等内容。本步骤规定，凡是属于同一个解码单元的多个 RTP 数据报文都将拥有相同的时间戳，但是序列号必须依据原始的数据位移相继递增。图 3 证明了这一点。这样，在客户端，就能够很容易的组装相应的数据报文成为一个个完整的解码单元，实现播放。

流媒体分布式控制文件定义步骤

一个电影源文件将会被分割成多个分片文件，并且存储在多个节点上。当产生了一个客户对某电影播放请求时，如何保证视频服务器能够统一调度该电影的所有分片，而不会有遗漏等错误是一个至关重要的问题。这就需要存储系统提供有效的电影分片控制信息来协助视频服务器完成一个电影的完整播放。

在本步骤中，包括电影所有分片的播放长度、序号、存储位置、

所需带宽等基本信息将会以一种固定的方式被提取出来。本系统规定每个分片的播放称其为一个基本播放任务（Playing Task）；那么，一个完整的电影在播放时，将会拥有一个基本播放任务列表。视频服务器只需要把这个基本播放任务列表读入内存，就可以实现准确无误的调度。

本步骤所涉及的分布式控制文件主要有两个，一个是 Index 文件，一个是 SDP 文件。Index 文件由一个播放任务列表、电影资源文件名、电影资源空间大小、电影资源时间长度、电影资源分片数目、电影资源热点度组成；而该播放任务列表包含了所有的播放任务项目；每一个播放任务项目由本任务的起始时间点、结束时间点、起始序列号、结束序列号以及本任务对应的分片文件所在的节点机的 IP 地址集合组成。如图 4 所示。

SDP 信息文件则用于描述电影资源的基本信息、客户端解码前的准备信息。它是由媒体类型编号、电影资源包含的媒体流的数目、电影资源的时间长度、客户会话的唯一表示码组成。如图 5 所示。

分片文件定义步骤

本步骤的核心文件是分片文件。分片文件的结构是单独定义的一套标准。依据该标准，本系统的分片执行模块能够对任何媒体格式的文件进行准确无误的切割分片，最终使得本系统能够脱离于具体媒体格式而运行。

根据本方法的设计，分片文件主要是由以下几个部分组成。

① 分片文件头部

② 媒体流信息头

③ 媒体流数据包

其中，②与③成组出现。如果本分片文件有 2 个流，则它包含两组②与③的配对。图 6 从总体简明地描述了分片文件的逻辑结构。

下面详细地描述这三个部分的结构。

(1) 分片文件头部

分片文件头部描述本分片的基本信息，它由本分片的序列号、本分片时间长度、本分片包含的媒体流的个数、本分片所需要的平均网

络带宽以及本分片文件的版本号。其中，所需要的平均网络带宽用于在视频服务器启动之后估计网络带宽的利用率，预先分配带宽；放置带宽的浪费或者饥饿。图 7 显示了分片文件头部的一些数据结构的定义。

(2) 媒体流信息头

媒体流信息头部描述了该媒体流的基本信息。媒体流指的是视频流、音频流或者系统流。当然，两个压缩码率不同的视频流也视为不同的媒体流。该部分由以下信息组成：媒体流的标志（用于解码器的识别）、媒体流的播放时间长度、媒体流的压缩码率、媒体流的数据起始位置（为文件读取接口提供媒体数据的位移）。图 8 显示了媒体流信息头部数据结构的详细定义。

(3) 媒体流的数据包

在流媒体网络数据报文定义模块中，已经详细描述了流媒体网络报文的结构。这里，媒体流的真正数据包是对流媒体网络数据报文的进一步封装而得到的。从图 3 中可以发现，序列号只是 16 位无符号整型数，它所能表示的范围有限，因此当出现较大的序列号时，只能够回绕。时间戳是 32 位无符号的整型，它是与本媒体流的特征速率相乘得到的一个数值；显然当出现大的时间戳时，也必须回绕。但究其本质，媒体流的数据实质上是由流媒体网络数据报文组成，因此，当要实现对媒体数据的随机访问时，可以回绕的时间戳与序列号将难以起到正确的作用。故本系统将对流媒体网络数据报文进行封装，将媒体流 ID 标志、在整个媒体流的数据包中的序列号以及实际的播放时间点（从零计算起）封装起来。这三个数据项可以方便地实现对媒体流数据的随机访问。图 9 显示了媒体流数据结构。

媒体源文件信息获取步骤

本方法所定义的一系列的数据结构都与本步骤有关。本步骤的执行，是为了获取媒体源文件的基本信息，这些基本信息包括：源文件的空间大小、源文件的媒体流个数、每个媒体流的播放时间长度、每个媒体流的媒体格式类型 ID、整个源文件总的播放时间。这些数据均可以从媒体源文件之中获得。但由于具体的媒体格式的不同，所以，

对于每一种媒体格式都要求写一个信息获取子模块。

下面以 Mpeg-1 系统流为例描述信息获取的流程。

Mpeg-1 媒体格式有一些特点。首先，这种文件的组织有着严格的二进制的规定。媒体源文件之中有一个系统头部，读取这个系统头部，就可以得到这个媒体源文件的媒体格式、媒体流的种类，以及压缩比率。

然后，分析媒体格式的媒体数据部分。Mpeg-1 媒体格式的媒体数据是以 Pack 的方式组织起来的，每个 Pack 的大小基本固定，而且都是可以自由解码的，因此，它就是我们在前面所定义的独立解码单元。每个 Pack 有一个固定的信息头部，它记载了本 Pack 的播放时间点，因此可以得到播放时间，从前向后分析，直到分析最后一个 Pack 的头部，我们就可以得到这个媒体源文件的总的播放时间。

这样，就完成了媒体源文件信息的获取工作。

分片文件放置策略定义步骤

分片文件需要存储在所有的存储节点上，当然是按照一定的规则、策略进行的。考虑放置策略时，既要兼顾热点电影的高访问频率，又要考虑各个存储节点的负载均衡性和系统的容错性。本模块包含了一个数据放置策略选择项、电影资源热点级别选择项以及可供调用的数据放置实现算法。

本系统提出一种用户有限干预的放置策略。首先提供传统的放置策略：RR（Round Robin）循环轮转策略；其次，由用户给定一个电影的热点性参数，以此来决定该电影的每个分片的复制份数。系统根据用户提供的信息即可实现电影的所有分片文件的分布式存储。

给定热点性参数 Hot，其数据结构的定义如下：

```
Hot_Level{
        First_Level; //第一级，该媒体文件的任何分片都没有复制存储;
        Second_Level;//第二级，该媒体文件的每一个分片都有一个复制;
        Third_Level; //第三级，该媒体文件的每一个分片
```

都有两个复制；

Top_Level;　//最高级，该媒体文件的每一个分片都有三个复制；

};

int Hot;

循环轮转的算法描述如下：

定义 N 个存储节点机为：Host[I], I=1, 2, …N；某一个电影的所有 M 个分片为：Clips[I], I=1, 2, …M；第一个复制序列为：Clips_One[I], I=1, 2…M；第二个复制序列为：Clips_Two[I], I=1, 2…M ；第三个复制序列为：Clips_Three[I], I=1, 2, …M。

当热点参数为第一级时，

第 I 个分片的存储位置为 Host[a]：Host[a]= I mod N;

当热点参数为第二级时，

第 I 个分片的存储位置为 Host[a]：Host[a]= I mod N;

第一个复制序列中第 J 个复制分片的存储位置为 Host[b]:Host[b]=(J mod N) +1;

当热点参数为第三级时，

第 I 个分片的存储位置为 Host[a]：Host[a]= I mod N;

第一个复制序列中第 J 个分片的存储位置为 Host[b]：Host[b]=(J mod N) +1;

第二个复制序列中第 K 个分片的存储位置为 Host[c]：Host[c]=(K mod N) +2;

当热点参数为最高级时，

第 I 个分片的存储位置为 Host[a]：Host[a]= I mod N;

第一个复制序列中第 J 个分片的存储位置为 Host[b]：Host[b]=(J mod N) +1;

第二个复制序列中第 K 个分片的存储位置为 Host[c]：Host[c]=(K mod N) +2;

第三个复制序列中第 L 个分片的存储位置为 Host[d]：Host[d]=(L mod N) +3;

流媒体文件分析与分片任务列表生成步骤

该步骤通过获得分片文件的放置策略信息，得到用户定义的分片文件放置要求；同时还要分析媒体源文件，以寻找每一个分片文件在源文件中的空间时间偏移量以及网络报文的序列号的范围，在此基础上可以得到分片任务列表。流媒体文件分析与分片任务列表生成步骤的处理流程包含一个操作序列，这个序列主要由以下步骤组成：第一步，依据已经获取的媒体文件基本信息寻找 MPEG 标准所定义的 PACK 标志；第二步，处理该 PACK 单元的数据。即分析该单元数据的时间戳、全局序列号；第三步，把此时间戳与每一个分片文件的规定时间长度进行比较，相等时，即可以写一个分片列表项目到 Index 文件中；否则继续处理下一个 PACK 数据单元。如图 10 所示。

分片任务列表的数据结构如图 11 所示。从图中可以发现，任务列表详细的记载了一个媒体源文件的每一个分片任务的基本情况，包括在源文件中的空间偏移量、开始播放时间点、解码单元的序号以及有关于调度这个分片任务的参数，如开始调度时间、本分片任务完成时间、完成百分比、完成的结果是否成功等。

分片任务执行步骤

该步骤依据分片任务列表，通过创建多线程同时执行多个分片任务，它包括一个操作序列，这个序列描述如下：第一步，读取分片控制 Index 文件，取得分片数目，以此数目建立多个处理线程；第二步，继续读取 Index 文件，获得一个分片任务列表，把这个列表每个表项的内容交给相应的线程，从而建立每个分片任务。第三步，对于任意一个分片任务，在电影资源文件中定位到规定的空间位置，寻找 PACK 标志，当遇到此标志时，就读出这个数据单元，并按照网络流媒体报文定义模块的说明，分割成若干个网络流媒体的数据包，并写入相应的分片文件；循环操作，直至所有相关的数据单元都完全读完；这样，一个分片任务就最终完成了。在这个分片任务执行中，定时监测它的完成百分比，显示给用户浏览；并记载它的调度时间信息并判断是否成功，以返回给用户决策。

分片任务执行模块的处理流程如图 12 所示。

分片传递存储步骤

该步骤的任务是把经过以上的分片任务执行步骤所得到的分片文件存储到响应的节点上去。首先依据分片文件放置策略定义步骤所定义的位置要求获得节点的网络地址，然后把得到的分片文件传递存储上去。

说 明 书 附 图

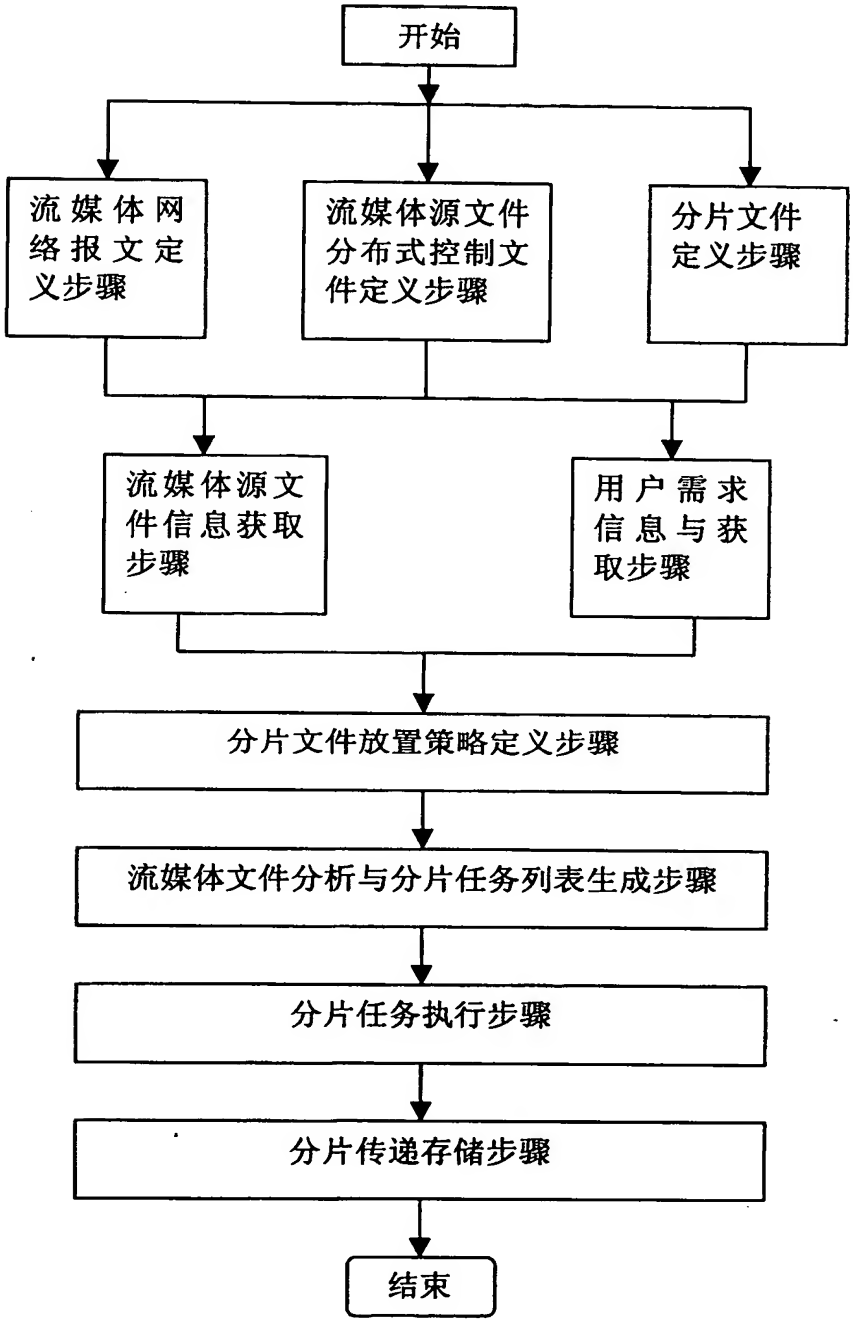
开始
流媒体网络报文定义步骤
流媒体源文件分布式控制文件定义步骤
分片文件定义步骤
流媒体源文件信息获取步骤
用户需求信息与获取步骤
分片文件放置策略定义步骤
流媒体文件分析与分片任务列表生成步骤
分片任务执行步骤
分片传递存储步骤
结束

图 1

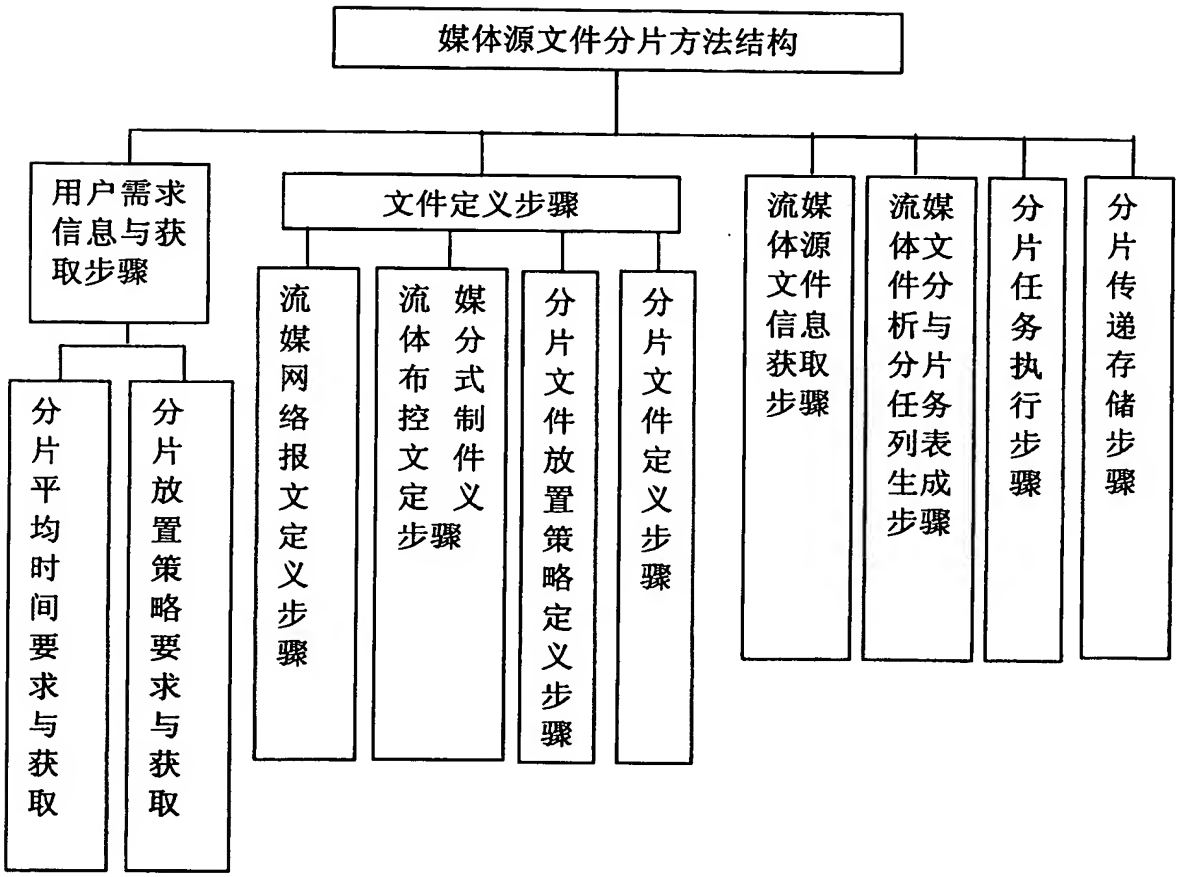
媒体源文件分片方法结构
用户需求信息与获取步骤
分片平均时间要求与获取
分片放置策略要求与获取
文件定义步骤
流媒网络报文定义步骤
流媒体布控文件定义步骤
分片文件放置策略定义步骤
分片文件定义步骤
流媒体源文件信息获取步骤
流媒体文件分析与分片任务表成列生步骤
分片任务执行步骤
分片传递存储步骤

图 2

一个完整的解码单元：

[redacted]

流媒体网络报文：

媒体类型头 (UInt16)	序列号 (UInt16)	时间戳 (UInt32)	同步源 (UInt32)	主体数据 (UInt8[])
T	N	t	SSRC	[redacted]
T	N+1	t	SSRC	[redacted]
T	N+2	t	SSRC	[redacted]
T	N+3	t	SSRC	[redacted]

图 3

```
//分片文件信息，基本播放任务
typedef struct Clip
{
   //本分片及其复制分片所处的主机地址序列
   UInt32      fHostIP[MAX_REPLICA_NUM];
   //该分片的空间大小，以字节为单位；
   UInt32      fFileSize;
   //该分片的起始播放时间，以秒为单位；
   Float64     fStartTime;
   //该分片的截止播放时间，以秒为单位
   Float64     fEndTime;
   //该分片的第一个网络数据报文的序列号；
   UInt32      fStartPacketIndex;
   //该分片的最后一个网络数据报文的序列号；
   UInt32      fEndPacketIndex;
}Clip;

//一个电影源文件的分片列表，基本播放任务列表
typedef struct ClipTable
{
   //该电影源文件空间大小，以字节计算；
   UInt32            fFileSize;
   //该电影的热点参数；
   int               fHot;
   //该电影源文件的名字长度；
   UInt8             fNameLen;
   //该电影源文件的分片个数
   UInt8             fNumber;
   //该电影源文件的名字；
   char              *fName;
   //指向每个分片的指针；
   Clip              *fIndex;
}ClipTable;
```

图 4

```
b=AS:1383
a=range:npt=0-  46.57500
m=OTHER 0 RTP/AVP 96
b=AS:1383
a=rtpmap:96 MP1S/90000
a=control:trackID=2
```

图 5

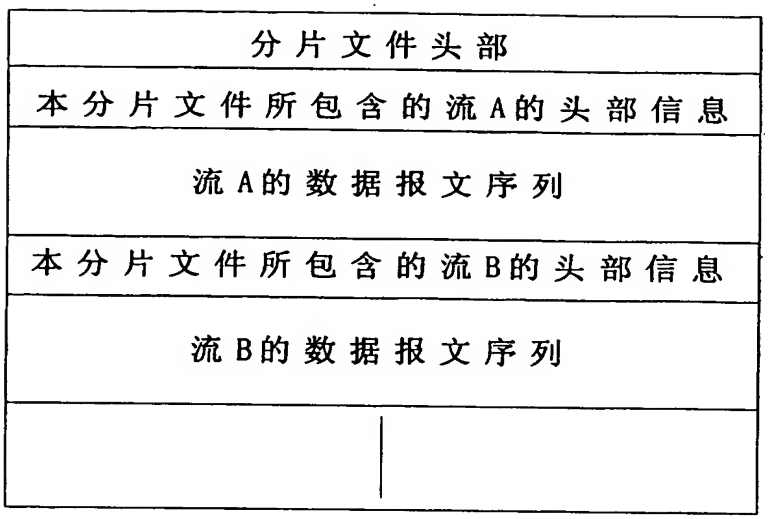

图6

```
/*
    分片文件头部
*/

typedef struct FileHeader
{
    // 分片文件ID
    UInt16   fSplit_ID;
    // 当前分片工具的版本号
    UInt32 fVersion;
    //本分片总的播放时间
    Float64  fMovieDuration;
    //本分片所包含的媒体流的个数
    UInt32 fNumTracks;
    //本分片所需的平均带宽
    Float64   fBandWidth;
}FileHeader
```

图7

```
/*
    分片文件中媒体流信息头部
*/
typedef struct TrackHeader
{
    //本媒体流的ID标志
    UInt8     fTrackID;
    //本媒体流总的播放时间
    Float64   fTrackDuration;
    //本媒体流的压缩码率
    Float64   fCompressRatio;
    //本媒体流的媒体数据的起始位置
    UInt32    fMediaPosition;
}FileHeader
```

图 8

媒体流的数据包：

媒体流的ID (UInt8)	序列号 (UInt32)	播放时间点 (Float64)	网络报文长度 (UInt16)	网络包文 (UInt8[])

图 9

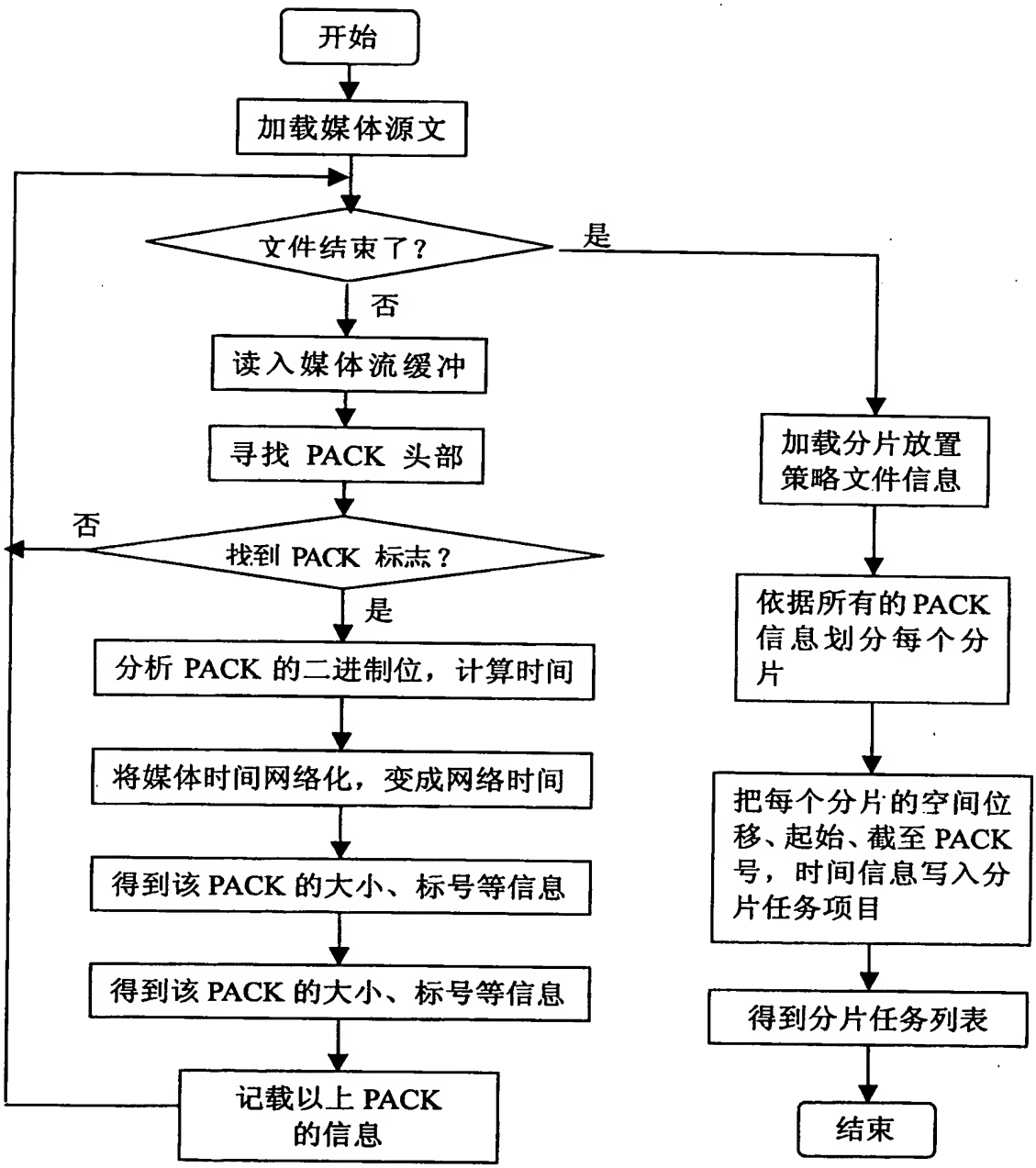
开始
加载媒体源文
文件结束了？
是
否
读入媒体流缓冲
寻找 PACK 头部
否
找到 PACK 标志？
是
分析 PACK 的二进制位，计算时间
将媒体时间网络化，变成网络时间
得到该 PACK 的大小、标号等信息
得到该 PACK 的大小、标号等信息
记载以上 PACK 的信息
加载分片放置策略文件信息
依据所有的PACK信息划分每个分片
把每个分片的空间位移、起始、截至 PACK 号，时间信息写入分片任务项目
得到分片任务列表
结束

图 10

```
//每一个分片任务
typedef struct Each_Task_Info
{
   //本分片开始时间，以秒为单位;
   Float64 fStartTime;
   //本分片包含的起始Pack序号
   UInt32  fStartPackIndex;
   //本分片包含的空间起始偏移
   UInt32  fStartPosition;
   //本分片包含的空间终止偏移
   UInt32  fEndPosition;
   //本分片任务所操作的分片编号
   UInt32  fIndex;
   //本分片任务的完成情况,百分比值
   Float64 fWorkingProcessing;
   //本分片任务的调度开始时间
   Time    fSchedule_Start_Time;
   //本分片任务完成的总体时间，以秒为单位
   Float64 fSchedule_Total_Time;
   //本分片任务完成与否;
   UInt8   fSucceed;
}Each_Task_Info;

//总体分片任务列表;
typedef struct Task_Info
{
  //本列表中的任务数目;
  UInt8            fNumber;
  //本列表所对应的媒体源文件的句柄
  int              fSourceFile;
  //所有分片任务项目
  Each_Task_Info  *fIndex[MAX_SPLIT_NUMBER];
}Task_Info;
```

图 11

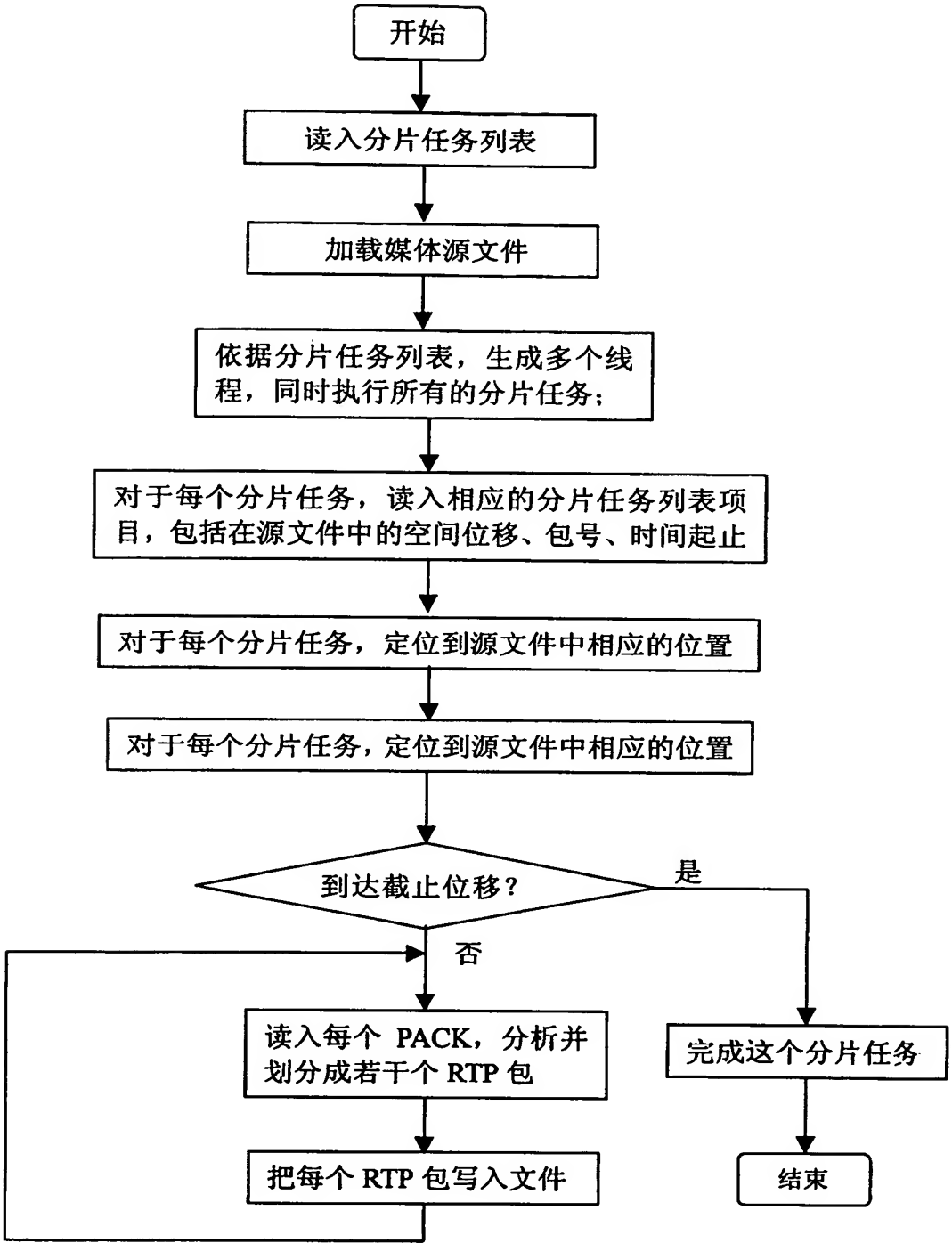
开始
读入分片任务列表
加载媒体源文件
依据分片任务列表，生成多个线程，同时执行所有的分片任务；
对于每个分片任务，读入相应的分片任务列表项目，包括在源文件中的空间位移、包号、时间起止
对于每个分片任务，定位到源文件中相应的位置
对于每个分片任务，定位到源文件中相应的位置
到达截止位移？
是
否
读入每个 PACK，分析并划分成若干个 RTP 包
把每个 RTP 包写入文件
完成这个分片任务
结束

图 12